AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application.

1. (Currently amended) A method for mapping a descriptive language including a data description having a structure complexity into an object oriented programming language~~data presentation~~, comprising the steps of:

receiving the data description;

identifying a complex-type element in the data description; and

creating an executable object oriented class that is independently executable in a run-time environment and that corresponds ~~corresponding~~ to the identified complex-type element, wherein the class includes an internal static class, wherein the internal static class corresponds to the structure complexity of the data description.

2. (Previously amended) The method as recited in claim 1, wherein receiving the data description comprises receiving an XML Schema.

3. (Previously amended) The method as recited in claim 1, further comprising validating the data description.

4. (Previously amended) The method as recited in claim 3, wherein said validating further includes using a object finite state machine including a current state to verify a mutator method call against the current state of the object, wherein the data description is invalid when the mutator method call is initiated before the current state is complete.

5.   (Previously amended) The method as recited in claim 3, wherein said validating includes:

sending a request including said data description to a remote server; and

receiving a validity determination as to said data description.

6.   (Previously amended) The method as recited in claim 3, wherein said validating includes:

reading said data description into a set of valid descriptor classes; and

creating a set of objects out of the data description wherein the occurrence of an object reflects validity.

7.   (Original) The method as recited in claim 1, wherein said creating includes a set of object oriented classes selected from the group consisting of: Java, C++ and Smalltalk.

8.   (Original) The method as recited in claim 1, wherein said creating includes representing a naming space with said internal static class to provide an implementation of said structure complexity.

9.   (Currently amended) A method for mapping a Schema including a structural complexity into an executable object oriented programming language, wherein the object oriented programming language provides a one to one correspondence between the structural complexity of the Schema and the functionality of the object oriented programming language,

the method comprising ~~the steps of~~:

receiving said Schema;

validating said Schema;

creating a set of <u>executable</u> object oriented classes including a set of internal static classes

<u>that are independently executable in a run-time environment and</u> to provide a mapping of the

Schema into the object oriented language;

creating an instance corresponding to the object oriented classes;

compiling the instance to provide an object oriented <u>programming</u> code.

10.   (Original) The method as recited in claim 9, wherein said validating includes

using a object finite state machine including a current state to verify a function call against the

current state of the object, wherein the Schema is invalid when the function call is initiated

before the current state is complete.

11.   (Previously amended) The method as recited in claim 9, wherein said validating

includes:

sending a request including said Schema to a remote server; and

receiving a validity determination as to said Schema.

12.   (Previously amended) The method as recited in claim 9, wherein said validating

includes:

reading said Schema into a set of valid descriptor classes; and

creating an instance of a compiler class wherein the compiler class is described in the

Schema.

13.   (Previously amended) The method as recited in claim 9, wherein the set of object

oriented classes comprises code selected from the group consisting of: Java, C ++, and Smalltalk.

14.   (Original) The method as recited in claim 9, wherein said creating an instance

includes representing a naming space with the internal static class to provide an implementation

of said structural complexity.

15.   (Currently amended) A computer readable medium containing programming

which when executed performs the following:

receiving a data description;

identifying a complex-type element in the data description; and

creating an executable object oriented class that is independently executable in a run-time

environment and that corresponds corresponding to the identified complex-type element,

wherein the class includes an internal static class, wherein the internal static class corresponds to

a structure complexity of the data description.

16.   (Previously currently amended) The medium as recited in claim 15, wherein

receiving the data description comprises receiving an XML Schema.

17.   (Previously amended) The medium as recited in claim 15, further comprising

validating the data description.

18.   (Previously amended) The medium as recited in claim 17, wherein said validating

procedure further includes using an object finite state machine including a current state to verify

a mutator method call against the current state of the object, wherein the data description is

invalid when the mutator method call is initiated before the current state is complete.

19.   (Previously amended) The medium as recited in claim 17, wherein said validating

procedure includes:

sending a request including said data description to a remote server; and

receiving a validity determination as to said data description.

20.   (Previously amended) The medium as recited in claim 17, wherein said validating

procedure includes:

reading said data description into a set of valid descriptor classes; and

creating a set of objects out of the data description wherein the occurrence of an object

reflects validity.

21.   (Original) The medium as recited in claim 15, wherein said creating procedure

includes a set of object oriented classes selected from the group consisting of: Java, C ++ and

Smalltalk.

22.   (Original) The medium as recited in claim 15, wherein said creating procedure

includes representing a naming space with said internal static class to provide an implementation

of said structure complexity.

23.   (Currently amended) A computer readable medium containing programming for

mapping a Schema including a structural complexity into an executable object oriented

programming language, wherein the object oriented programming language provides a one to

one correspondence between the structural complexity of the Schema and the functionality of the

object oriented programming language which when executed performs the following procedures

comprising:

receiving said Schema;

validating said Schema;

creating a set of executable object oriented classes including a set of internal static classes

that are independently executable in a run-time environment and to provide a mapping of the

Schema into the object oriented language;

creating an instance corresponding to the object oriented classes; and

compiling the instance to provide an object oriented programming code.

24.   (Previously amended) The medium as recited in claim 23, wherein said validating

procedure includes using an object finite state machine including a current state to verify a

function call against the current state of the object, wherein the Schema is invalid when the

function call is initiated before the current state is complete.

25.   (Previously amended) The medium as recited in claim 23, wherein said validating

procedure includes:

sending a request including said Schema to a remote server; and

receiving a validity determination as to said Schema.

26.   (Previously amended) The medium as recited in claim 23, wherein said validating

procedure includes:

reading said Schema into a set of valid descriptor classes; and

creating an instance of a compiler class wherein the compiler class is described in the

Schema.

27.   (Previously amended) The medium as recited in claim 23, wherein the set of

object oriented classes comprises code selected from the group consisting of: Java, C ++, and

Smalltalk.

28.   (Original) The medium as recited in claim 23, wherein said creating an instance

procedure includes representing a naming space with the internal static class to provide an

implementation of said structural complexity.

29.   (Currently amended) An apparatus for mapping a descriptive language including a

data description having a structure complexity into an object oriented <u>programming language,</u>

~~data presentation~~ comprising:

means for receiving the data description;

means for identifying a complex-type element in the data description; and

means for creating an executable object oriented class <u>that is independently executable in</u>

a run-time environment and that corresponds ~~corresponding~~ to the identified complex-type

element, wherein the class includes an internal static class, wherein the internal static class

corresponds to the structure complexity of the data description.

30.   (Previously amended) The apparatus as recited in claim 29, wherein said data

description comprises an XML Schema.

31.   (Previously amended) The apparatus as recited in claim 29, further comprising

means for validating a data description.

32.   (Previously amended) The apparatus as recited in claim 29, wherein said

validating means further includes a object finite state machine including a current state to verify

a mutator method call against the current state of the object, wherein the data description is

invalid when the mutator method call is initiated before the current state is complete.

33.   (Previously amended) The apparatus as recited in claim 29, wherein said

validating means includes a web browser operable to send a request including said data

description to a remote server for validation.

34.   (Previously amended) The apparatus as recited in claim 29, wherein said

validating means includes:

means for reading said data description into a set of valid descriptor classes; and

means for creating a set of objects out of the data description wherein the occurrence of

an object reflects validity.

35.   (Previously amended) The apparatus as recited in claim 29, wherein said object

oriented classes comprise code selected from the group consisting of: Java, C++, and Smalltalk.

36.   (Original) The apparatus as recited in claim 29, wherein said creating means

includes means for representing a naming space with said internal static class to provide an

implementation of said structure complexity.